

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平6-223095

(43)公開日 平成6年(1994)8月12日

(51)Int.Cl.⁵

G 0 6 F 15/31

11/10

// H 0 3 M 13/00

識別記号

M 7343-5L

3 3 0 Q 7313-5B

8730-5J

庁内整理番号

F I

技術表示箇所

審査請求 未請求 請求項の数 6 O L (全 10 頁)

(21)出願番号 特願平5-9333

(22)出願日 平成5年(1993)1月22日

(71)出願人 000001007

キャノン株式会社

東京都大田区下丸子3丁目30番2号

(72)発明者 岩村 恵市

東京都大田区下丸子3丁目30番2号キャノ

ン株式会社内

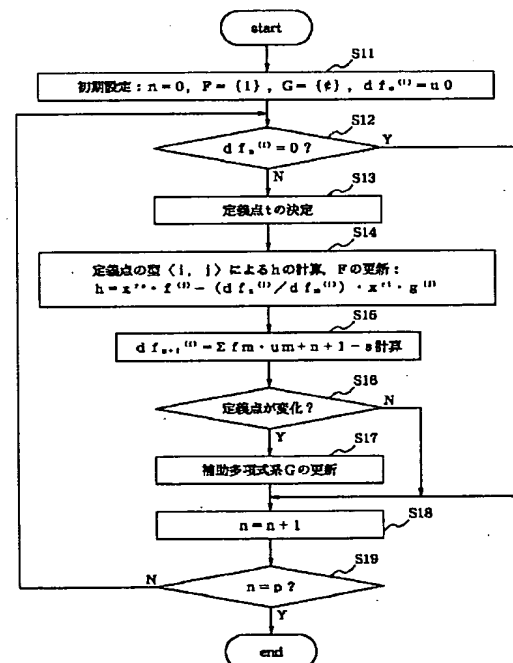
(74)代理人 弁理士 丸島 儀一

(54)【発明の名称】 多項式系導出装置及びその方法

(57)【要約】

【目的】 代数幾何符号の復号において、与えられた多次元配列を生成する最小多項式系を高速に求める。

【構成】 与えられた多次元配列 u を格納する配列記憶手段と、求める多項式系 F 、当該多項式系 F と異なる多項式系 G をそれぞれ格納するための第1、第2多項式記憶手段と、前記第1、第2多項式記憶手段にそれぞれ記憶された多項式 $f^{(1)}$ 、 $g^{(j)}$ に対して、演算 $f^{(k)} = z^{rs} \cdot f^{(1)} - (df_n^{(1)}/df_m^{(j)}) \cdot z^{rt} \cdot g^{(j)}$ を行う第1演算手段と、該演算された多項式 $f^{(k)} = \sum f_m^{(k)} \cdot z_m$ と前記記憶された多次元配列 u に対して演算 $df_{n+1}^{(k)} = \sum f_m^{(k)} \cdot u_{m+n+1-s}$ を行なう第2演算手段と、前記第1、及び第2の多項式記憶手段に対するアクセス及び該アクセスのアドレスを多項式 $f^{(k)}$ の次数に依存させて並列に制御する制御手段とを具える。



【特許請求の範囲】

【請求項1】 与えられた多次元配列を生成する最小多項式系を求める装置において、

与えられた多次元配列 u を格納する配列記憶手段と、
求める多項式系 F を格納するための第1多項式記憶手段と、

当該多項式系 F と異なる多項式系 G を格納するための第2多項式記憶手段と、

前記第1多項式記憶手段に記憶された多項式 $f^{(i)}$ 、前記第2多項式記憶手段に記憶された多項式 $g^{(j)}$ に対して、演算

$$f^{(k)} = z^{rs} \cdot f^{(i)} - (df_n^{(i)} / df_m^{(j)}) \cdot z^{rt} \cdot g^{(j)}$$

を行う第1演算手段と、

該第1演算手段によって演算された多項式 $f^{(k)} = \sum f_m^{(k)} \cdot z_m$ と前記配列記憶手段に記憶された多次元配列 u に対して演算

$$df_{n+1}^{(k)} = \sum f_m^{(k)} \cdot um+n+1-s$$

を行なう第2演算手段と、

前記第1、及び第2の多項式記憶手段に対するアクセス及び該アクセスのアドレスを多項式 $f^{(k)}$ の次数に依存させて並列に制御する制御手段とを有し、前記第1及び第2の演算手段による演算を並列に実行することを特徴とする多項式系導出装置。

【請求項2】 前記第1あるいは第2の多項式記憶手段が、複数のメモリを具え、複数の多項式を、多項式毎に異なるメモリに格納することを特徴とする請求項1に記載の多項式系導出装置。

【請求項3】 前記第1あるいは第2の多項式記憶手段が、複数のメモリを具え、複数の多項式を、変数及び次数毎に異なるメモリに格納することを特徴とする請求項1に記載の多項式系導出装置。

【請求項4】 前記制御手段が、複数の多項式を多項式毎に異なるアドレスに格納するように、前記第1あるいは第2の多項式記憶手段を制御することを特徴とする請求項1に記載の多項式系導出装置。

【請求項5】 前記制御手段が、複数の多項式を変数及び次数毎に異なるアドレスに格納するように、前記第1あるいは第2の多項式記憶手段を制御することを特徴とする請求項1に記載の多項式系導出装置。

【請求項6】 与えられた多次元配列を生成する最小多項式系を求める方法において、

与えられた多次元配列 u を配列メモリに格納し、
求める多項式系 F を格納するための第1多項式メモリに

記憶された多項式 $f^{(i)}$ 、当該多項式系 F と異なる多項式系 G を格納するための第2多項式メモリに記憶された $g^{(j)}$ に対して、第1の演算

$$f^{(k)} = z^{rs} \cdot f^{(i)} - (df_n^{(i)} / df_m^{(j)}) \cdot z^{rt} \cdot g^{(j)}$$

を実行し、

該第1の演算によって求められた多項式 $f^{(k)} = \sum f_m^{(k)} \cdot z_m$ と前記配列メモリに記憶された多次元配列 u

に対して第2の演算

$$df_{n+1}^{(k)} = \sum f_m^{(k)} \cdot um+n+1-s$$

を実行し、

前記第1、及び第2の多項式メモリに対するアクセス及び該アクセスのアドレスを多項式 $f^{(k)}$ の次数に依存させて並列に制御し、前記第1及び第2の演算を並列に実行することを特徴とする多項式系導出方法。

10 【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、ディジタル通信系及びディジタル記憶系において通信路または記憶媒体で受けた誤りを、誤り訂正符号を用いて受信側で自動的に訂正する誤り訂正に関し、特に誤り訂正符号として代数幾何符号を用いる場合の復号において、与えられた多次元配列を生成する最小多項式系を求める多項式系導出装置及びその方法に関する。

【0002】

20 【従来の技術】従来、ディジタル通信系及びディジタル記憶系において通信路または記憶媒体で受けた誤りを、受信側で自動的に訂正する誤り訂正符号として、リード・ソロモン(RS)符号[1]やBCH符号[1]がよく知られ、コンパクト・ディスクや衛星通信等において実際に使用されている。そのRS符号やBCH符号の復号法として、バーレカンプ・マッセイ(BM)法がよく知られ、装置化されている。また、最近では代数曲線論を駆使した代数幾何符号[1]~[4]と呼ばれる符号の研究が盛んに行われている。代数幾何符号は前述のRS符号やBCH符号をその1クラスとして含む非常に適用範囲の広い符号系であり、従来の符号よりもよい性質を持つ新符号が存在することが徐々に明らかになってきている[5]

30 [6]。その復号法として阪田によって提案されたアルゴリズムが知られている[7]。このアルゴリズムは2次元BM法と呼ばれ、RS符号の復号に用いられるBM法(以後、1次元BM法と呼ぶ)の拡張になっていることが知られている。また、BM法は阪田によって多次元にまで拡張され[8]、図7のような構造を持つアルゴリズムとして表される。(ただし、定義点の決定法、及び r_s 、 r_t 等は文献[7][8]参照。)

【0003】

40 【発明が解決しようとする課題】図7の n は全順序(文献[7][8]参照)と呼ばれる順序づけによって更新される。図7において2)の $df_n^{(i)}$ の計算と4)の h による $f_n^{(i)}$ の更新は逐次的に行われることが多かった。なぜならば、点 n における2)の $df_n^{(i)}$ は点 $n-1$ における4)の h を用いて計算され、点 n における4)の h による $f_n^{(i)}$ の更新は点 n の2)の $df_n^{(i)}$ を用いて計算されるためである。この場合、図8に示すように処理

50 時間に無駄が生じるために効率的ではなかった。

3

【0004】一方、RS符号やBCH符号の復号法である1次元BM法の1次元変数についての処理を逐次的に行うことを利用して2)と4)の処理を並列に実行する手法が文献[9]に提案されている。しかし、この手法は段数固定のシフトレジスタを用いているために1回の $f_n^{(i)}$ の更新(1サイクル)に必ず $t+1$ (t は多項式 $f_n^{(i)}$ の最終点($n=p$:図7参照)における定義点の最大値)の処理クロックが必要であった。従って、文献[9]の装置は最大の定義点とならない $f_n^{(i)}$ の途中演算($n < p$)において無駄な処理クロックが必要であり、更に多次元のBM法を考慮していないために、次の点において文献[9]の手法は多次元BM法には不向きであった。

【0005】1) 1次元BM法では多項式系 F, G に属す多項式は1つであるが、多次元BM法では多項式系 F, G に属す多項式は複数存在する。

【0006】2) 1次元BM法では多項式系 F, G に属す多項式の次数と与えられる配列の配置は1次元(1変数)であるので、高次から順に1次元状のメモリ(シフトレジスタ等)に各係数を格納できるが、多次元BM法の多項式の次数と配列の配置は1次元ではないので1次元状のメモリでは効率的な処理ができない。

【0007】3) 1次元BM法では多項式系 F, G に属す多項式は1つであるので、多項式間の並列性は意味が無いが、多次元BM法では多項式系 F, G に属す多項式は複数存在するので、多項式間の演算も並列化できる。

【0008】4) 1次元BM法では多項式系 F, G に属す多項式の変数は1つであるので、1つの変数についての演算を並列に行うと2)と4)の処理は並列化できないが、多次元BM法では多項式系 F, G に属す多項式の変数は複数存在するので、1つの変数についての演算を並列に行っても2)と4)の処理を並列化できる。

【0009】そこで、本発明では上述の欠点を除去し、多次元BM法に対しても2)と4)の処理を効率的に並列処理するアルゴリズム及び、それを実現する装置を提案することを目的とする。これによって、1次元BM法を含む多次元BM法の処理時間が短縮され、高速な処理が実現できる。

【0010】[参考文献]

- [1] 今井: “符号理論”, 電子情報通信学会
- [2] V. D. Goppa: “Codes on algebraic curves”, Soviet Math. Dokl., 24, pp. 170-172, 1981.
- [3] V. D. Goppa: “Algebraico-geometric codes”, Math. U.S.S.R. Izvestiya, vol. 21, no. 1, pp. 75-91, 1983.
- [4] V. D. Goppa: “Geometry and Codes”, Kluwer Academic Publishers, 1988.
- [5] M. A. Tsfasman and S. G. Vladut: “Algebraic-geometric codes”, Kluwer Academic Publishers, 1991.

4

[6] 三浦: “ある平面曲線上の代数曲線符号”, Preprint.

[7] 阪田: “与えられた2次元配列を生成する2次元線形帰還シフトレジスタの合成”, 信学論(A), vol. J-70A, pp. 903-910, 1987.

[8] S. Sakata: “Extension of the Berlekamp-Massey algorithm to N dimensions,” Information and Computation, vol. 84, pp. 207-239, 1990

[9] Youzhi Xu: “Contributions to the Decoding of Reed-Solomon and Related Codes,” Linkoping Studies in Science and Technology. Dissertations No. 257, 1991.

【0011】

【課題を解決するための手段】上記課題を解決するために、本発明では、与えられた多次元配列を生成する最小多項式系を求める装置に、与えられた多次元配列 u を格納する配列記憶手段と、求める多項式系 F を格納するための第1多項式記憶手段と、当該多項式系 F と異なる多項式系 G を格納するための第2多項式記憶手段と、前記第1多項式記憶手段に記憶された多項式 $f^{(1)}$ 、前記第2多項式記憶手段に記憶された多項式 $g^{(j)}$ に対して、演算 $f^{(k)} = z^{rs} \cdot f^{(1)} - (df_n^{(1)}/df_n^{(j)}) \cdot z^{rt} \cdot g^{(j)}$ を行う第1演算手段と、該第1演算手段によって演算された多項式 $f^{(k)} = \sum f_n^{(k)} \cdot z_m$ と前記配列記憶手段に記憶された多次元配列 u に対して演算 $df_{n+1}^{(k)} = \sum f_n^{(k)} \cdot u_{m+n+1-s}$ を行なう第2演算手段と、前記第1、及び第2の多項式記憶手段に対するアクセス及び該アクセスのアドレスを多項式 $f^{(k)}$ の次数に依存させて並列に制御する制御手段とを具える。

【0012】

【作用】与えられた多次元配列 u を配列記憶手段に格納し、求める多項式系 F を第1多項式記憶手段に、当該多項式系 F と異なる多項式系 G を第2多項式記憶手段にそれぞれ格納するようにし、第1演算手段によって前記第1多項式記憶手段に記憶された多項式 $f^{(1)}$ 、前記第2多項式記憶手段に記憶された多項式 $g^{(j)}$ に対して、演算 $f^{(k)} = z^{rs} \cdot f^{(1)} - (df_n^{(1)}/df_n^{(j)}) \cdot z^{rt} \cdot g^{(j)}$ を実行し、該第1演算手段によって演算された多項式 $f^{(k)} = \sum f_n^{(k)} \cdot z_m$ と前記配列記憶手段に記憶された多次元配列 u に対して、第2演算手段によって演算 $df_{n+1}^{(k)} = \sum f_n^{(k)} \cdot u_{m+n+1-s}$ を実行し、制御手段が前記第1、及び第2の多項式記憶手段に対するアクセス及び該アクセスのアドレスを多項式 $f^{(k)}$ の次数に依存させて並列に制御することで、前記第1及び第2の演算手段による演算を並列に実行する。

【0013】

【実施例】

【説明の概要】N次元BM法において2点間の加減算は各次元の変数について独立に行われる。そのために、N次元BM法においてはN次元の変数を独立に扱うN次元

5

状のメモリを用いた方が効率的な処理を行うことができる。

【0014】更に、 $N-1$ 次元の変数についての演算を同時に行っても、残りの変数についての処理が逐次的に行われるので、2)と4)の処理を並列化できる。 $N-1$ 次元の変数についての演算を同時に行うためには、その変数について各次数毎に独立したメモリを用いて同時にアクセスすればよい。従って、本発明では多項式及び各変数の次数毎に独立のメモリをメモリのアドレス分割または複数のメモリによって実現し、効率的に多次元BM法の処理を行うアルゴリズム及び装置を提案する。

【0015】図7に対応する本発明によるアルゴリズムを図1に、それによる動作を図2に示す(図2(a)～(c)は実施例1～3に対応する)。図1において2)と4)及び5)の処理は並列に行われている。

【0016】〔実施例1〕まず1次元BM法について考える。図3において u は与えられた1次元配列 u (アドレス i には u_i が設定)を格納するメモリ、 f は u を生成する最小多項式 f (初期値は1)を格納するメモリ、 g は f の補助多項式 g (初期値は0)を格納するメモリであり、そのアドレスは外部から制御される。ここでは1次元BM法を考えているのでメモリに格納される多項式は1つであり、その多項式の最高次数を s とした場合 $s-i$ 次の次数の係数はメモリのアドレス i に設定される。また、 df_n は $df_n^{(1)}$ を計算するためのレジスタ、 df_m は前の df_n の値を保持しておくレジスタ、 IN は逆数を出力する回路であり、 dd は $(df_n^{(1)} / df_m^{(j)})$ の値を保持するレジスタ(初期値は u_0)である。さらに、 SW は定義点に変化した場合のみメモリ f からの出力を選択するスイッチであり、その制御は外部から行われる。また、 \times 、 $+$ は各々乗算器、加算器を表す。

【0017】点 n において、多項式 f の定義点を $s^{(1)}$ 、多項式 g の定義点を $s^{(j)}$ とする。メモリ f 、 g は各々アドレス0から1ずつ順次加算され、アドレス $s^{(1)}$ 、 $s^{(j)}$ まで制御される。このとき、外部回路は定義点から計算される $rs + s^{(1)}$ と $rt + s^{(j)}$ の大小関係を比較し、大きい方のメモリから動作させはじめ、その差分のクロック数後には他方のメモリも動作させる。前述したように dd のレジスタは $(f_n^{(1)} / df_m^{(j)})$ の値を保持しているので、メモリ f には4)の h によって演算・更新された点 n における多項式 f の各係数がアドレス0から $s^{(k)}$ ($s^{(k)}$ は更新された多項式の定義点)へ順次入力される。このとき、更新された多項式 f の各係数に合わせて、メモリ u はアドレス $n+1$ から順次1ずつ減じてアドレス $n+1-s^{(k)}$ まで制御され、そこに蓄えている1次元配列 u の各要素を順次出力する。これによって、2)の点 $n+1$ における $df_{n+1}^{(k)}$ が4)の点 n における f と並列に計算される。このときレジスタ df_m には前の $df_n^{(1)}$ が入力され、 IN によるその逆数と $df_{n+1}^{(k)}$ との乗算を行うことにより、次の演

(4)

6

算における dd をレジスタに保持することができる。ただし、定義点に変化した場合には SW が開き、メモリ g にはメモリ f の出力がアドレス0からアドレス $s^{(1)}$ に対して順次入力され、多項式 g が更新される。従って、図3において多項式 f の1回の更新(1サイクル)には $s^{(k)}$ のクロック数が必要で、これを p 回繰り返すことにより u を生成する最小多項式 f を得ることができる。

【0018】 $s^{(k)}$ は $df_{n+1}^{(k)}$ 及び多項式 f の各係数を逐次処理する場合の最小クロック数である。従って、本発明は文献[9]の装置に比べて無駄な処理クロックがなくなり処理が高速化される。また、定義点の決定及び比較は高速性を要求されないのので、通常のCPUによって実現できる。従って、外部回路はCPUとアドレス制御を行うカウンタ回路によって簡単に実現できる。

【0019】〔実施例2〕実施例1に示したメモリのアドレス制御を行う外部回路は1次元BM法に対しては無駄な処理クロックを省くだけであったが、2次元以上のBM法に対しては前述の多次元状のメモリをアドレス制御によって実現し、効率的な多次元BM法を実行するために重要な意味を持つ。

【0020】まず簡単のために、2次元BM法について考える。図4において U は与えられた多次元配列 u を、 F は u を生成する複数の多項式 $f^{(i)}$ ($i=0, \dots, l-1$)を、 G は F の複数の補助多項式 $g^{(j)}$ ($j=0, \dots, l-2$)を格納するメモリである。 $df_n^{(i)}$ は $df_n^{(1)}$ を計算するためのレジスタ、 $df_m^{(j)}$ は前の $df_n^{(i)}$ の値を保持しておくレジスタであるが、多項式が複数あるのでこのレジスタも複数必要である。他の部品は実施例1と同じである。

【0021】ここで、 i を多項式 $f^{(i)}$ の次数、 j を y^j の次数、 k を $x^{s^{(1)}-k}$ の次数($s^{(1)}$ は多項式 $f^{(1)}$ の定義点)とすると、メモリ F 、 G は図5に示すようなアドレス分割が行え、そのアドレスは (i, j, k) と表現できる。ただし、図5における s_j は y^j に対する x の最大次数を表す。このとき、多項式系 F の中の1つの多項式 $f^{(1)}$ の定義点を $s^{(1)} = (s_1^{(1)}, s_2^{(1)})$ とし、 $a = (a_1, a_2)$ ($a \in \Sigma_0^{s^{(1)}} : \text{文献}[7][8]参照$)を変数とすると、多項式 $f^{(1)}$ はアドレス $(i, s_1^{(1)} - a_1, a_2)$ の位置に格納されている。また、多項式系 G の中の1つの多項式 $g^{(j)}$ の定義点を $s^{(j)} = (s_1^{(j)}, s_2^{(j)})$ とすると、多項式 $g^{(j)}$ は $a \in \Sigma_0^{s^{(j)}}$ に対してアドレス $(j, s_1^{(j)} - a_1, a_2)$ に格納されている。この場合、メモリ F 、 G に用いられる各 a は全順序に従う必要はなく、 a_1 と a_2 と独立に制御することができる。例えば、最初 $a_1 = a_2 = 0$ として a_1 を0から s_0 まで動作させた後、 a_2 を1繰上げ、 a_1 を再び0から s_1 まで動作させるような制御を行ってもよい。従って、 $a_1 a_2$ は通常のアップカウンタによって簡単に制御できる。また、メモリ U は点 $n = (n_1, n_2)$ の写像 u_n をアドレス (n_1, n_2) に割り付ければ、 (j, k) と表されるアドレス分割が

でき、メモリF、Gと同様に簡単なアドレス制御が行える。

【0022】従って、図4の装置は次のように動作させればよい。まず、外部回路は定義点から計算される $rs + s^{(i)}$ と $rt + s^{(j)}$ の全順序による大小関係と比較し、大きい方のメモリから動作させその多項式を出力する。その差分のクロック数後に他方のメモリも動作させ他方の多項式を出力する。ddのレジスタには($df_n^{(i)}/df_m^{(j)}$)の値が保持されているので、メモリFには4)のhによって演算・更新された定義点 $s^{(k)} = (s_1^{(k)}, s_2^{(k)})$ を持つ多項式 $f^{(k)}$ がアドレス($k, s_1^{(k)} - a_1, a_2$) ($a \in \Sigma_0^s(k)$)に対して順次入力されることがわかる。このとき、点nの全順序における次の点を $n+1 = n' = (n_1', n_2')$ とすると、更新された多項式 $f^{(k)}$ のアドレス内の変数aに合わせて、メモリUはアドレス($n_1' - a_1, n_2' - a_2$) ($a \in \Sigma_0^s(k)$)に蓄えている1次元配列uの各要素を出力する。これによって、2)の点n+1における $df_{n+1}^{(k)}$ が4)の点nにおける多項式 $f^{(k)}$ と並列に計算される。このときレジスタ $df_n^{(i)}$ には2)で計算された $df_{n+1}^{(k)}$ が、レジスタ df_m には前の $df_n^{(i)}$ が蓄えられ、後の演算において必要なときに用いられる。よって、次の演算において必要な($df_n^{(i)}/df_m^{(j)}$)の値がddのレジスタに保持され図1のアルゴリズムが連続して実行されていく。ただし、定義点に変化した場合にはSWが開き、メモリGにはメモリFのそのときの出力がアドレス($i, s_1^{(i)} - a_1, a_2$) ($a \in \Sigma_0^s(i)$)に対して順次入力され、多項式系Gが更新される。

【0023】従って、図4において1つの多項式 $f^{(i)}$ の更新(1サイクル)には $s^{(k)}$ クロックが必要である。多次元BM法ではステップS14で計算される多項式 $f^{(k)}$ とそれに用いられる多項式 $f^{(i)}$ 、 $g^{(j)}$ のk、i、jは定義点の型によって決定される。前述したように複数の多項式をアドレス分割して格納した場合、そのk、i、jはアドレスの指定によって順次選択できる。従って、点nにおいて指定された全ての多項式 $f^{(k)}$ を更新するためには、各 $s^{(k)}$ の和に相当するクロック数が必要になる。さらに、それを点pまで繰り返せば2次元配列uを生成する最小多項式系Fが得られる。

【0024】この多項式の選択も定義点を決定する外部回路によって実行できることは明かである。従って、アドレスを制御する外部回路を用いて複数の多項式をアドレス分割により選択することによって従来の文献[9]の装置で困難であった1)の問題を解決できる。さらに、問題2)に示された多次元の変数に関するメモリアクセスの複雑さの問題も図5に示すようにyの次数毎にアドレス分割を行うことによって容易になり解決できる。

【0025】また、各多項式は最小のクロック数 $s^{(k)}$ によって更新されるので、実施例1と同様に処理クロックを無駄にしない効率的な処理が行われていることは明

かである。

【0026】また、図4の回路がアドレスの分割数さえ増せば、N次元BM法に対しても有効であることも明かである。

【0027】〔実施例3〕実施例2は複数の多項式を1つのメモリのアドレス分割によって格納しているので、多項式及び各次数は1つしか選択できず問題3)、4)に示す多項式間及び変数毎の並列処理はなされていない。そこで、実施例3では問題3)、4)に示した多項式間及び変数毎の並列性を実現する装置を考える。

【0028】まず、多項式間の並列性を実現する本発明による実施例を図6に示す。図6においてメモリF、Gは複数の多項式を並列に格納する複数のメモリによって構成される。図6において太線で示される線は複数のメモリ数に応じた多重の信号線を意味し、+、×は多重の信号線に応じて並列に並べた複数の加算器及び乗算器を示す。また、レジスタ $df_n^{(i)}$ 、 $df_m^{(j)}$ 、dd及び逆数生成回路INも多重の信号線に応じて並列に並べられる。ただし、メモリUに格納されているuだけは1種類であり、複数の多項式 $f^{(i)}$ に対して共通であるので1本の線で表され、並列に並べられた乗算器に共通に入力される。

【0029】このとき、メモリF、Gのアドレスは各点nにおける最大の定義点を持つ多項式について制御すれば、他の多項式もそのアドレス制御によって制御できる。従って、点nにおける全ての多項式の更新にはその点における最大定義点である $s^{(k)}$ のクロック数でよく、それをp回繰り返すことによってuを生成する最小多項式系Fが得られる。

【0030】次に、変数毎の並列性を実現する装置も図6の実施例によって実現できる。この場合、実施例2の多項式のyの次数毎のアドレス分割を複数のメモリによって並列に格納すれば、1つの多項式内における変数yについての演算を並列に実現することができるので、多項式間だけでなく1つの多項式内の変数毎の並列処理を実現することができる。

【0031】また、図6の回路はメモリ数さえ増せば、N次元BM法に対しても有効であることも明らかである。

【0032】〔その他の実施例〕実施例3の装置のメモリ数は多項式分または次数分の数でなくても実施例2の場合と組み合わせて、準備できる複数のメモリをさらにアドレス分割して用いることもできる。

【0033】また、2)及び4)の処理を行う回路はCPU等によっても容易に構成することができ、本実施例に示した回路構成に制限されない。

【0034】また、ここに示したメモリの格納法及びアクセス法等は、多次元BM法に限らず、他の多次元配列及び多次元多項式生成法に関しても有効であることはいうまでもない。

【0035】

9

【発明の効果】本発明は1サイクルを逐次処理に必要な最小のクロック数 $s^{(k)}$ で実行できるので、無駄な処理クロックがなくなり処理が高速化される。

【0036】また、複数の多項式とその変数をアドレス分割を用いて制御することによって、多次元BM法にも簡単に対応することができる。

【0037】さらに、多次元BM法の複数の多項式を複数のメモリに分割して並列に制御することによって、複数の多項式を並列に処理することができ処理速度を向上させることができる。

【0038】さらに、多次元BM法の多項式内の変数の次数毎のアドレス分割を複数のメモリによって並列に格納することによって、1つの多項式内の変数毎の処理を並列に処理することができ処理速度をより向上させることができる。

【図面の簡単な説明】

【図1】本発明による多次元BM法のアルゴリズムである。

10

【図2】図1のアルゴリズムを実行した場合の動作図である。

【図3】図1のアルゴリズムの第1の実施例である。

【図4】図1のアルゴリズムの第2の実施例である。

【図5】メモリのアドレス分割の説明図である。

【図6】図1のアルゴリズムの第3の実施例である。

【図7】従来の多次元BM法のアルゴリズムである。

【図8】図7のアルゴリズムを実行した場合の動作図である。

10 【符号の説明】

＋ 加算器

× 乗算器

SW スイッチ

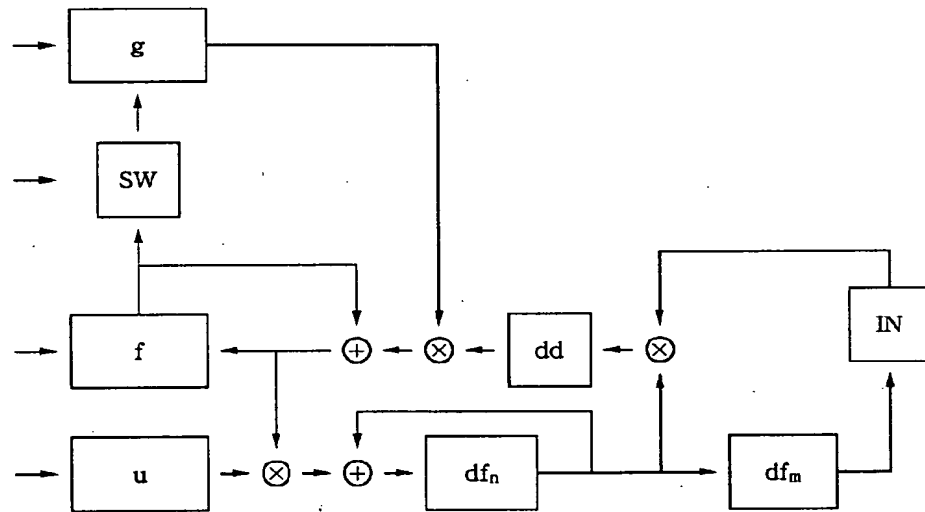
u, U 与えられた配列を格納するメモリ

f, g, F, G 多項式を格納するメモリ

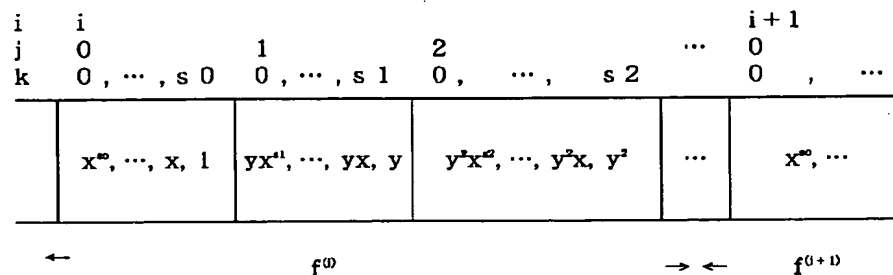
dd, df_n, df_m, df_n⁽¹⁾, df_m⁽¹⁾ レジスタ

IN 逆数を出力する回路

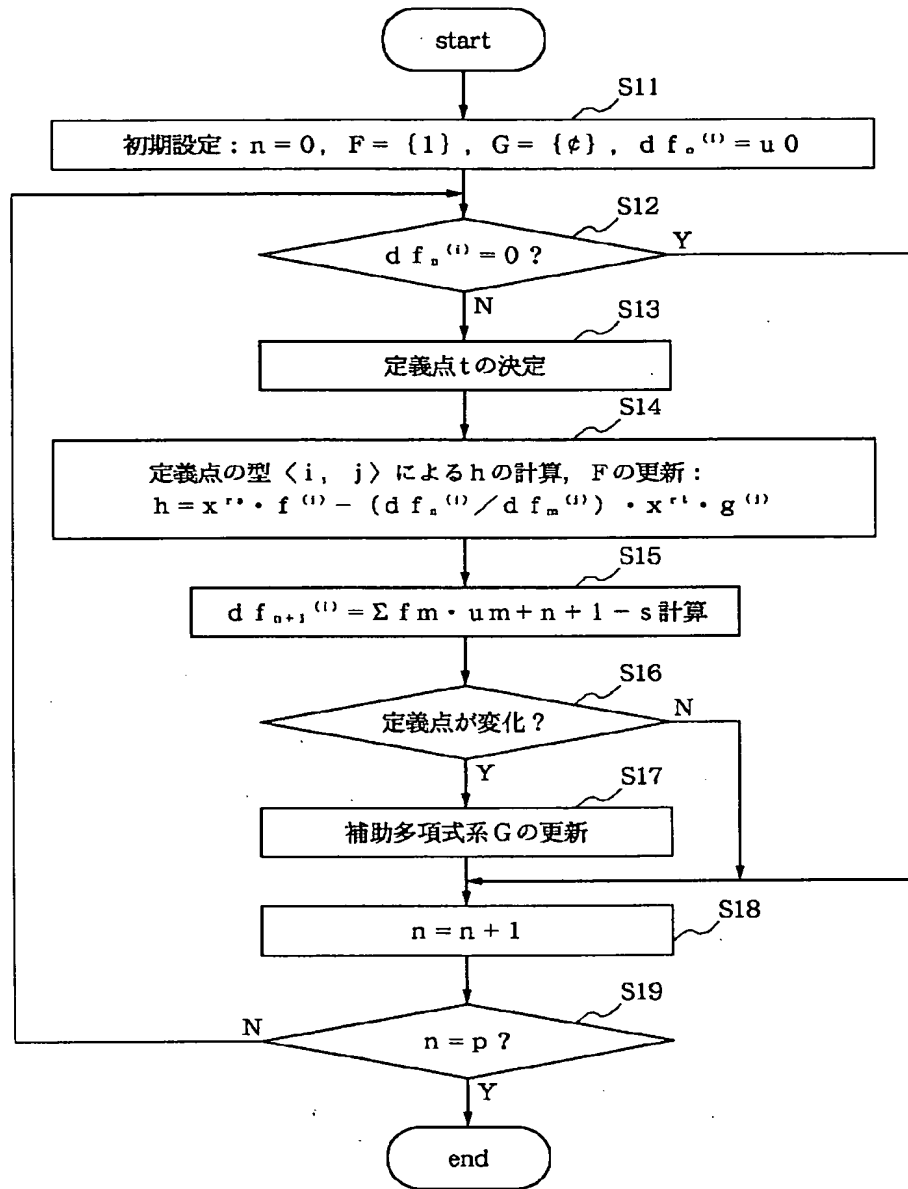
【図3】



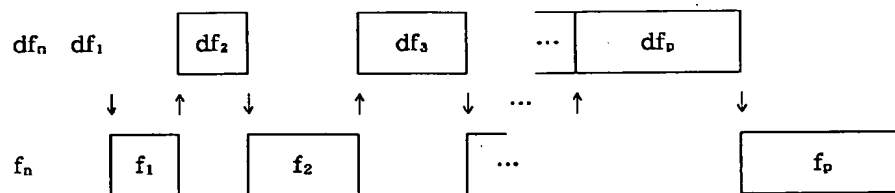
【図5】



【図1】

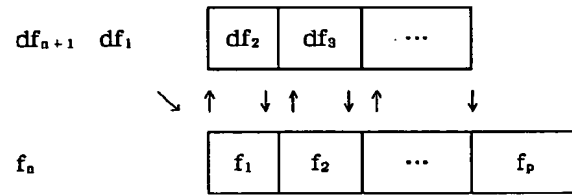


【図8】

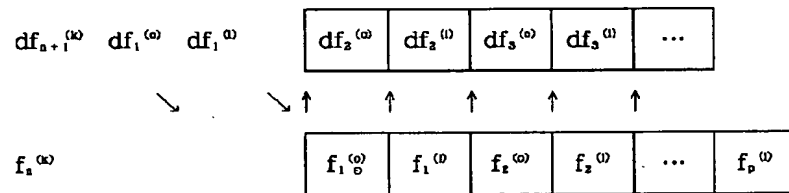


【図2】

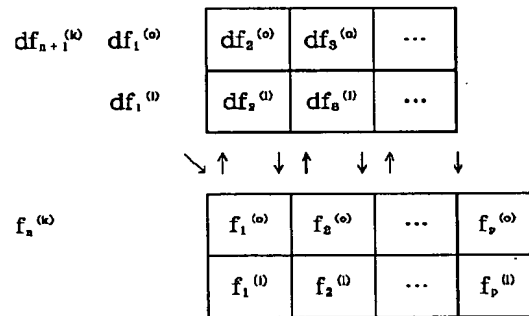
(a)



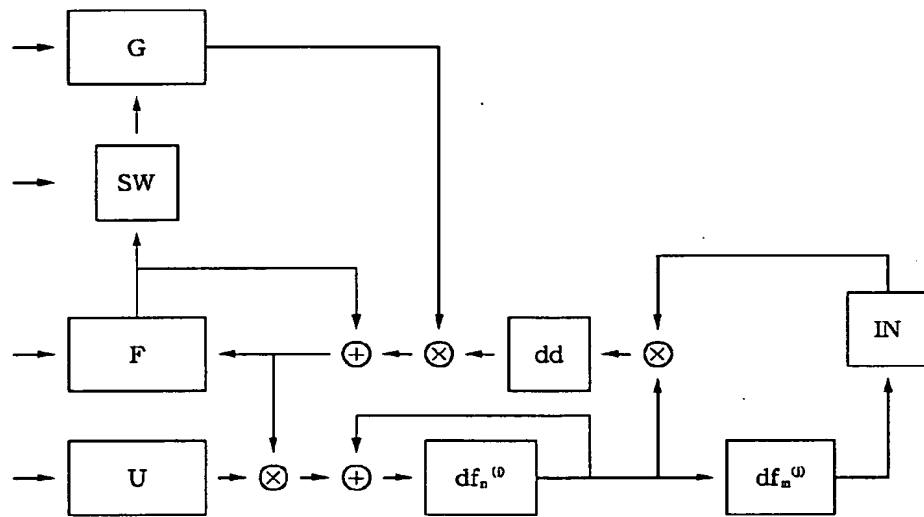
(b)



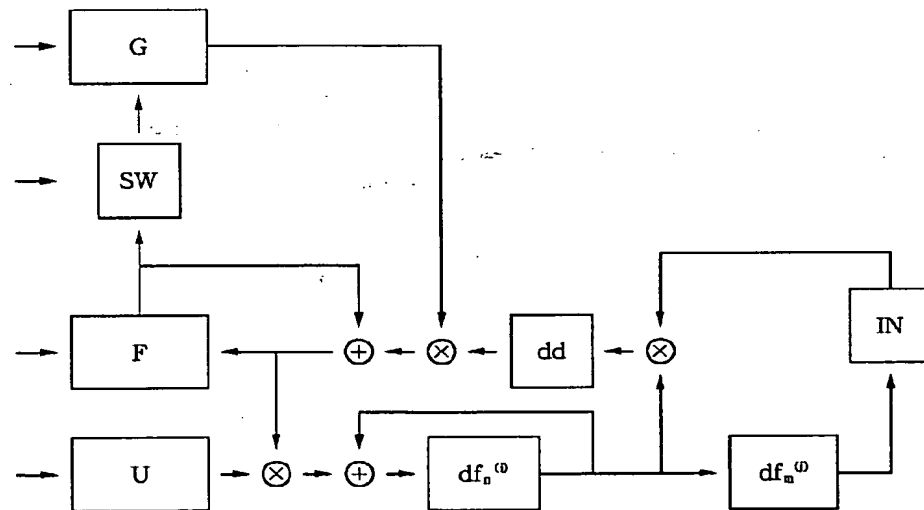
(c)



【図4】



【図6】



【図7】

